A Framework For Application Performance Understanding and Prediction

Laura Carrington Ph.D.
PMaC Lab
(Performance Modeling & Characterization)
at the San Diego Supercomputer Center

About us

- An NSF lab see www.sdsc.edu/PMaC
- The mission of the SDSC Performance Modeling and Characterization (PMaC) lab is to bring scientific rigor to the prediction of scientific application performance on current and projected HPC platforms.
- PMaC's goal is to predict the performance of applications more accurately than traditional benchmarking methods and more tractably than by traditional cycle-accurate performance simulations.
- Synergistic collaborations with DARPA HPCS PERCS, DOD HPCMO, DOE PERC

Why Performance Model?

- Performance models enable *understanding* of the factors that affect performance
  - Inform the tuning process (of application and machine)
  - Guide applications to the best machine
  - Enable applications driven architecture design
  - Extrapolate the performance of future systems

Overview

- Description of performance prediction framework and its uses:
  - Machine Profiles - MAPS
  - Application Signatures - MetaSim
  - Network simulator - DIMEMAS
- Performance prediction results
- Performance predictions to understand hardware upgrades and future architectures
- Performance predictions extrapolation to understand performance on large numbers of processors

The Performance Prediction Framework

- Parallel performance - 2 major factors:
  - Single processor performance
  - Use of the network

- 2 major components of the framework:
  - Single processor model

    Model of application's performance between communication events
  - Communication model (Network simulator)

    Model of application's communication events

The Performance Prediction Framework

- Both models based on simplicity and isolation:
  - Simplicity: start simple and only add complexity when needed to explain behavior
    - First assumption that a major performance factor for an application's is memory usage
  - Isolation: Collect each piece of the performance framework in isolation then combine pieces for performance prediction

Pieces of Performance Prediction Framework

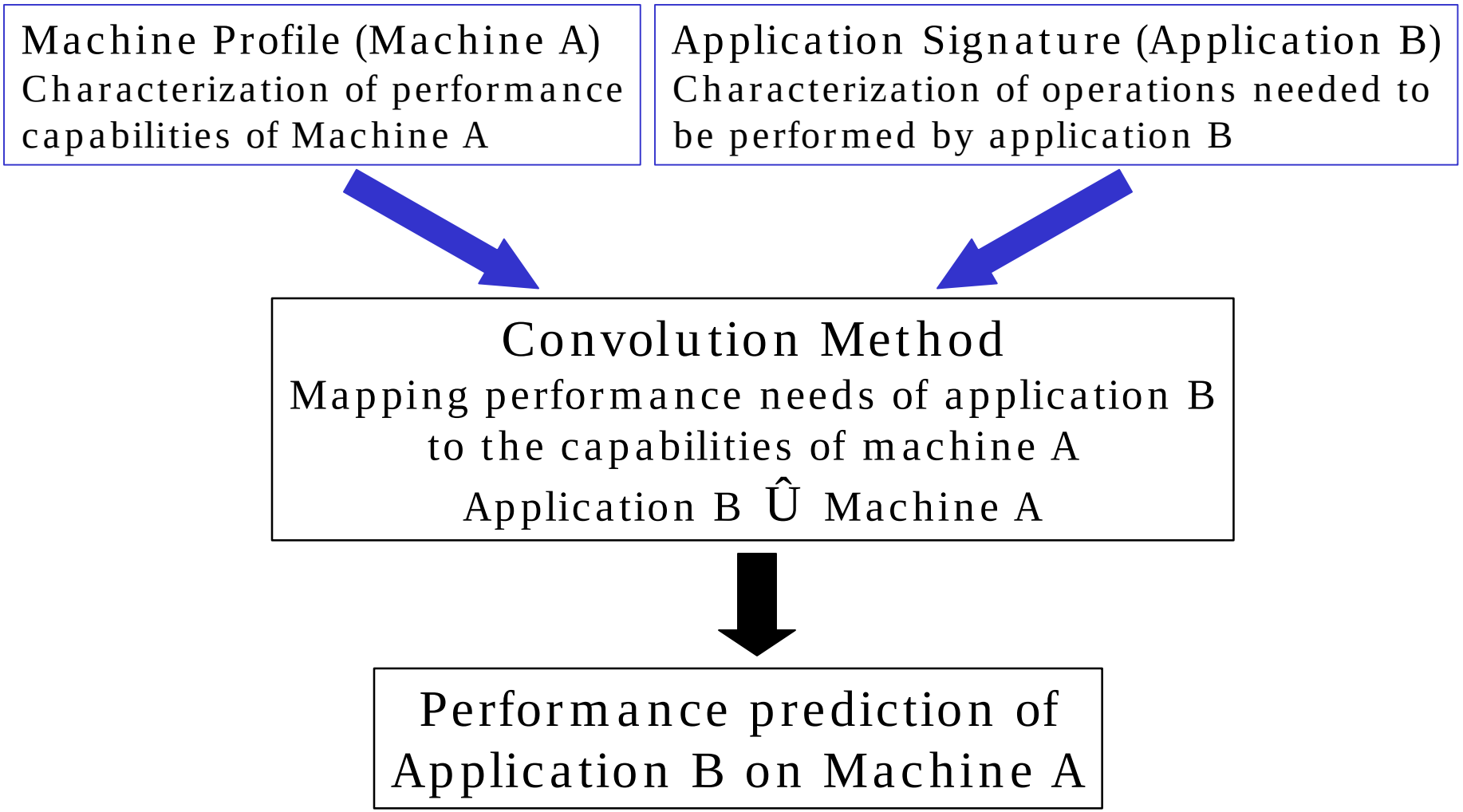
- **Machine Profile** - characterizations of the rates at which a machine can (or is projected to) carry out fundamental operations abstract from the particular application.
- **Application Signature** - detailed summaries of the fundamental operations to be carried out by the application independent of any particular machine.

Combine Machine Profile and Application Signature using:

- **Convolution Methods** - algebraic mappings of the Application Signatures on to the Machine profiles to arrive at a performance prediction.

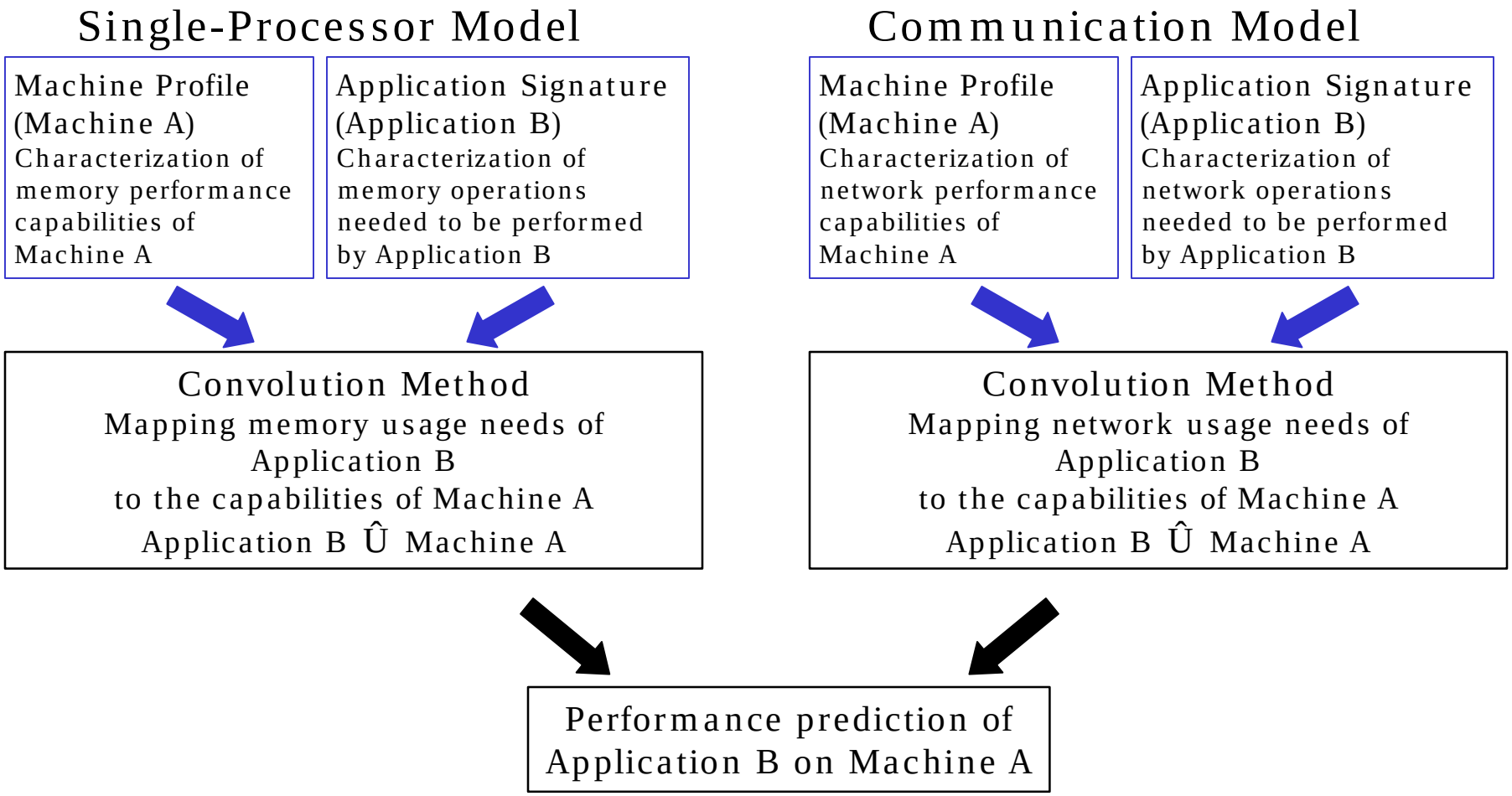
Pieces of Performance Prediction Framework

Single-Processor Prediction



Pieces of Performance Prediction Framework

Parallel Processor Prediction



Machine Profiles

Two components of Machine Profiles:

1. Single-processor model component
2. Communication model component

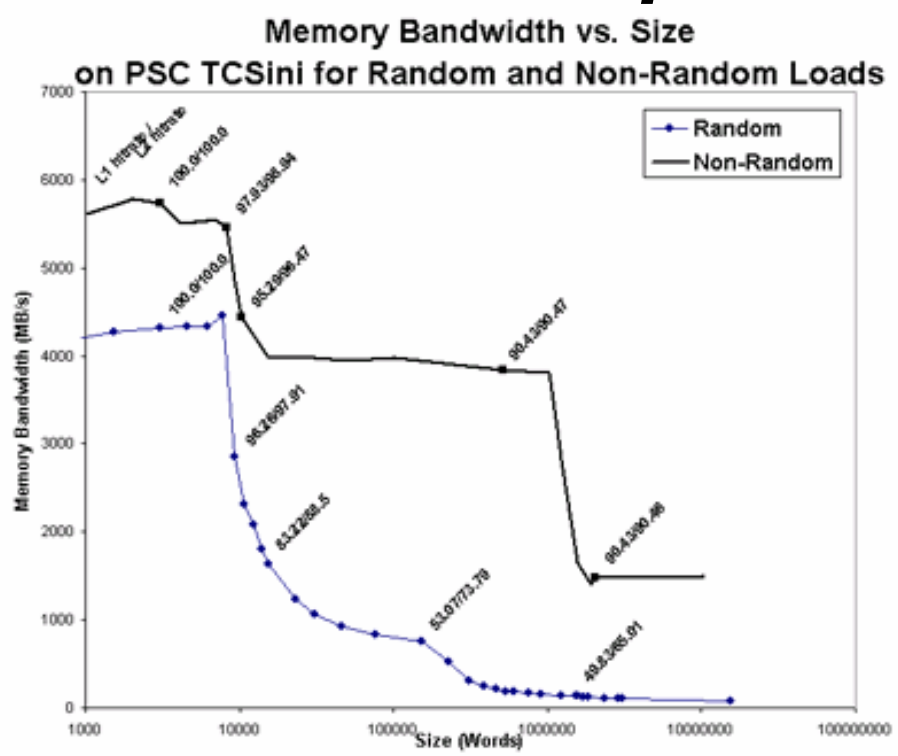
Single-processor model Machine Profile component:

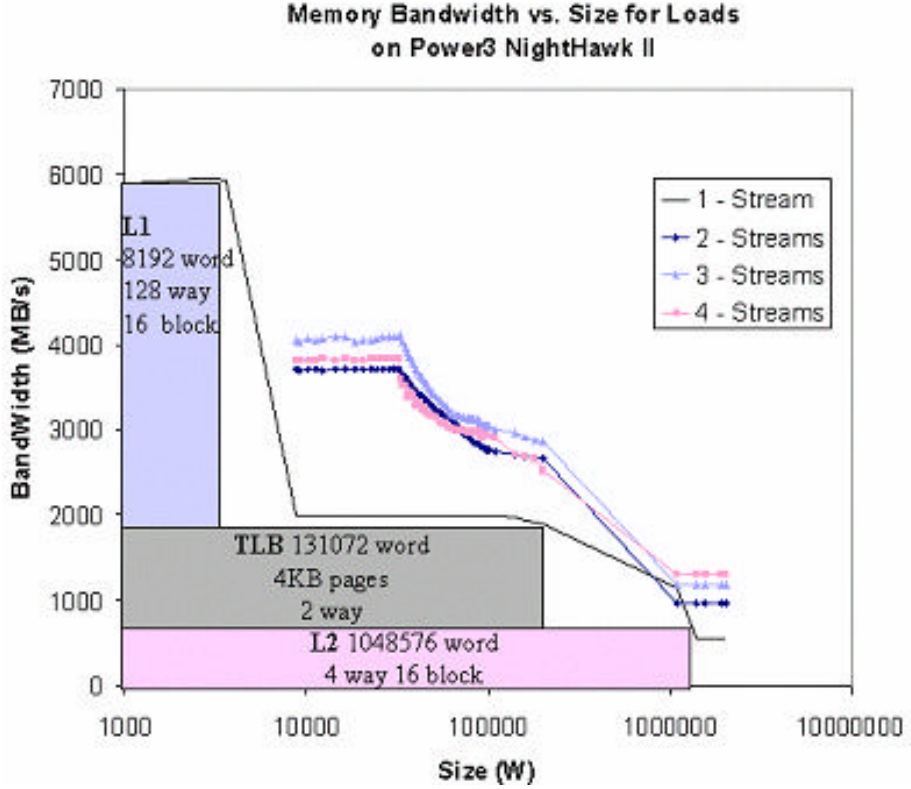
- Component based on machine's memory performance (First assumption: that a major performance factor for an application's is memory usage)

Communication model Machine Profile component:

- Component based on machines network performance

Machine Profiles – Single Processor Component – MAPS

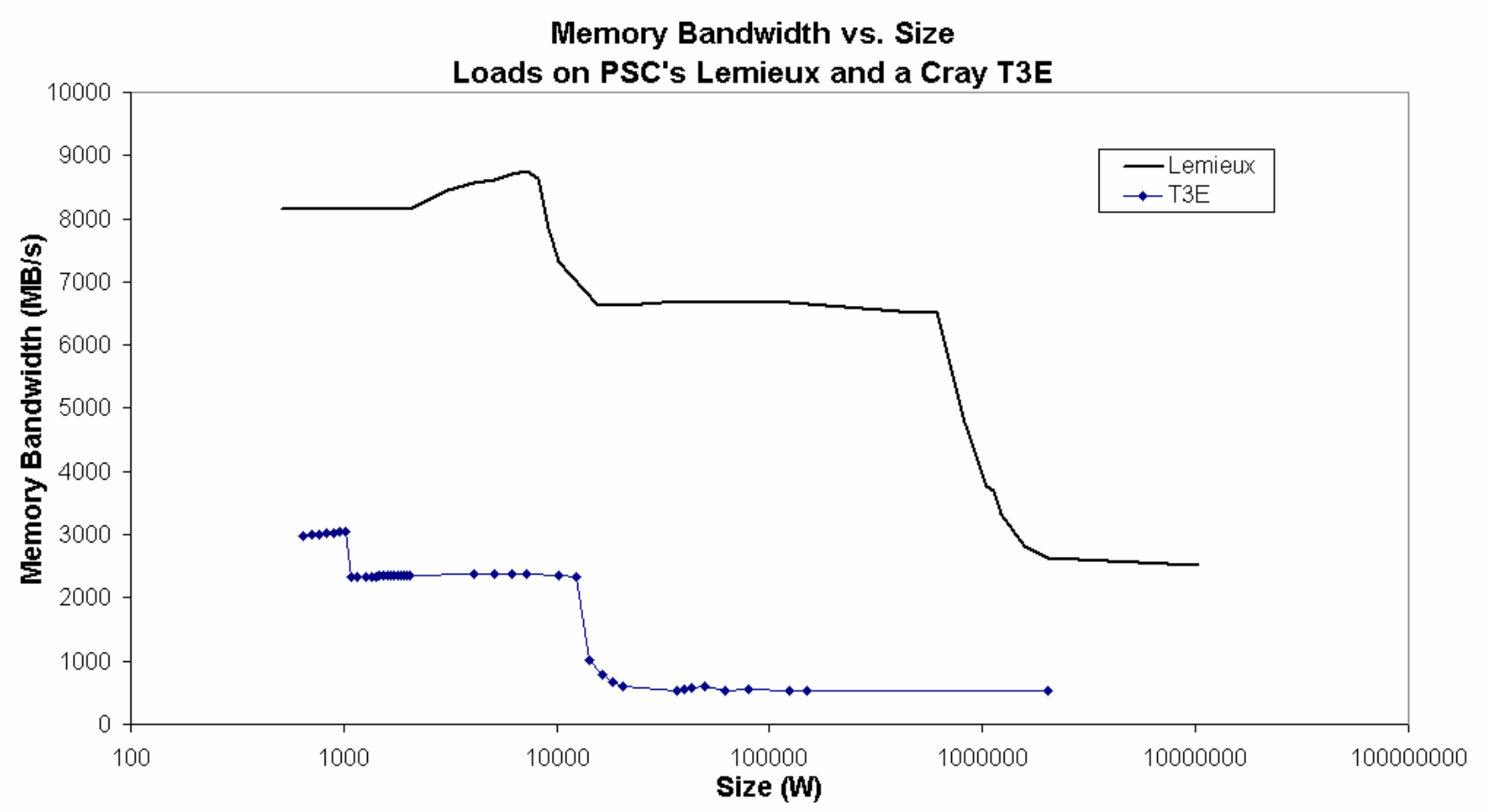




- Machine Profiles useful for :
  - revealing underlying capability of the machine
  - comparing machines
- Machine Profiles produced by:

  MAPS (*Memory Access Pattern Signature*) probe is available at www.sdsc.edu/PMaC

Comparing MAPS Profiles



Note: Size of L1 cache of Lemieux ≅ Size of L2 cache of T3E
Bandwidth ~8000MB/s Lemieux, ~2000MB/s T3E

Application Signatures

Two components of Application Signatures:

1. Single-processor model component
2. Communication model component

Single-processor model Application Signature component:

- Component based on application's memory usage (Memory trace tool, the MetaSim Tracer, traces an application's memory and floating-point usage)

Communication model Application Signature component:

- Component based on application's network usage (MPI trace tool, MPIDtrace, traces an application's communication events and CPU bursts between events)

Application Signature – MetaSim Tracer
Single-Processor component

MetaSim Tracer:
- Collects general application information
- Collects memory trace with user supplied memory parameters

General MetaSim trace collected on NPB CG class B on 32 CPUs

Basic Block #	# Inst.	# Memory References	% Total Mem. Ref.	Floating-Point Inst.	% FP Inst.	Random Ratio	Ratio of FP ops/ Mem. ops
373	2.1E+9	8.9E+8	0.22	8.2E+8	0.37	0.33	0.92
372	1.7E+9	8.6E+8	0.21	4.9E+8	0.22	0.37	0.57
371	1.3E+9	6.3E+8	0.15	3.6E+8	0.16	0.36	0.57
375	1.4E+9	5.0E+8	0.12	2.5E+8	0.11	0.35	0.50

Other information:
- Function name where basic-block is located
- Line number in code where basic-block is located

Application Signature – MetaSim Tracer
with User Memory Parameters

MetaSim trace collected on NPB CG class B on 32 CPUs with user supplied memory parameters for the IBM BlueHorizon

Basic Block #	% Total Mem. Ref.	Random Ratio	L1 Hit Rate	L2 Hit Rate	Data Set Location in Memory
373	0.22	0.33	92.16	99.98	L1 Cache
372	0.21	0.37	90.14	99.07	L1/L2 Cache
371	0.15	0.36	88.93	98.67	L1/L2 Cache
375	0.12	0.35	93.02	99.99	L1 Cache

MetaSim trace collected on NPB CG class B on 32 CPUs with user supplied memory parameters for the Cray T3E

Basic Block #	% Total Mem. Ref.	Random Ratio	L1 Hit Rate	L2 Hit Rate	Data Set Location in Memory
373	0.22	0.33	68.19	96.90	Main Memory
372	0.21	0.37	64.38	93.34	Main Memory
371	0.15	0.36	59.66	91.57	Main Memory
375	0.12	0.35	70.42	97.31	Main Memory

User supplied memory parameters:

- Sizes of L1/L2/L3 Cache
- Associativities of Caches

Application Signature – MetaSim Tracer

- Application Signature useful for :
  - revealing underlying implementation
  - comparing implementations
  - Identifying performance hotspots/bottlenecks in application

- Application Signature collected by:

  MetaSim Tracer is available at
  www.sdsc.edu/PMaC

  How to collect a memory trace with the MetaSim Tracer:

  % ATOM *executable* csim.anal.c csim.t3e.c

Download files at PMaC web site

Convolutions put the two together for Single-Processor Model

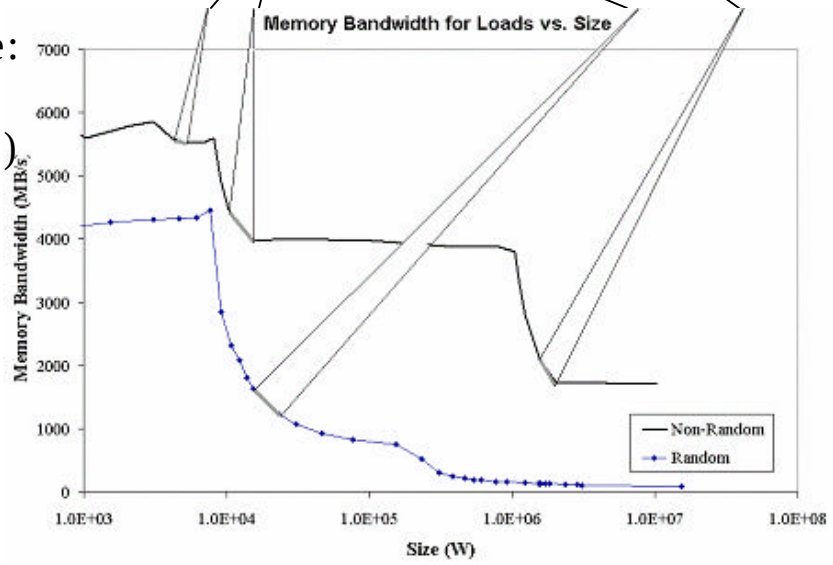
MetaSim trace collected on PETSc Matrix-Vector code 4 CPUs with user supplied memory parameters for PSC's TCSini

Procedure Name	% Mem. Ref.	Ratio Random	L1 Hit Rate	L2 Hit Rate	Data Set Location in Memory	Memory BW (MAPS)	Weighted BW
dgemv_n	0.9198	0.07	93.47	93.48	L1/L2 Cache	4166.0	3831.7
dgemv_n	0.0271	0.00	90.33	90.39	Main Memory	1809.2	49.1
dgemv_n	0.0232	0.00	94.81	99.89	L1 Cache	5561.3	129.3
MatSet.	0.0125	0.20	77.32	90.00	L2 Cache	1522.6	19.0

Single-processor or per-processor performance:
- Machine profile for processor (Machine A)
- Application Signature for application (App. #1)

The relative "per-processor" performance of App. #1 on Machine A is represented as the MetaSim Number=

$$\sum_{i=1}^{n} (\%\text{Mem. Ref. of } BB_i * \text{Mem. BW of } BB_i)$$



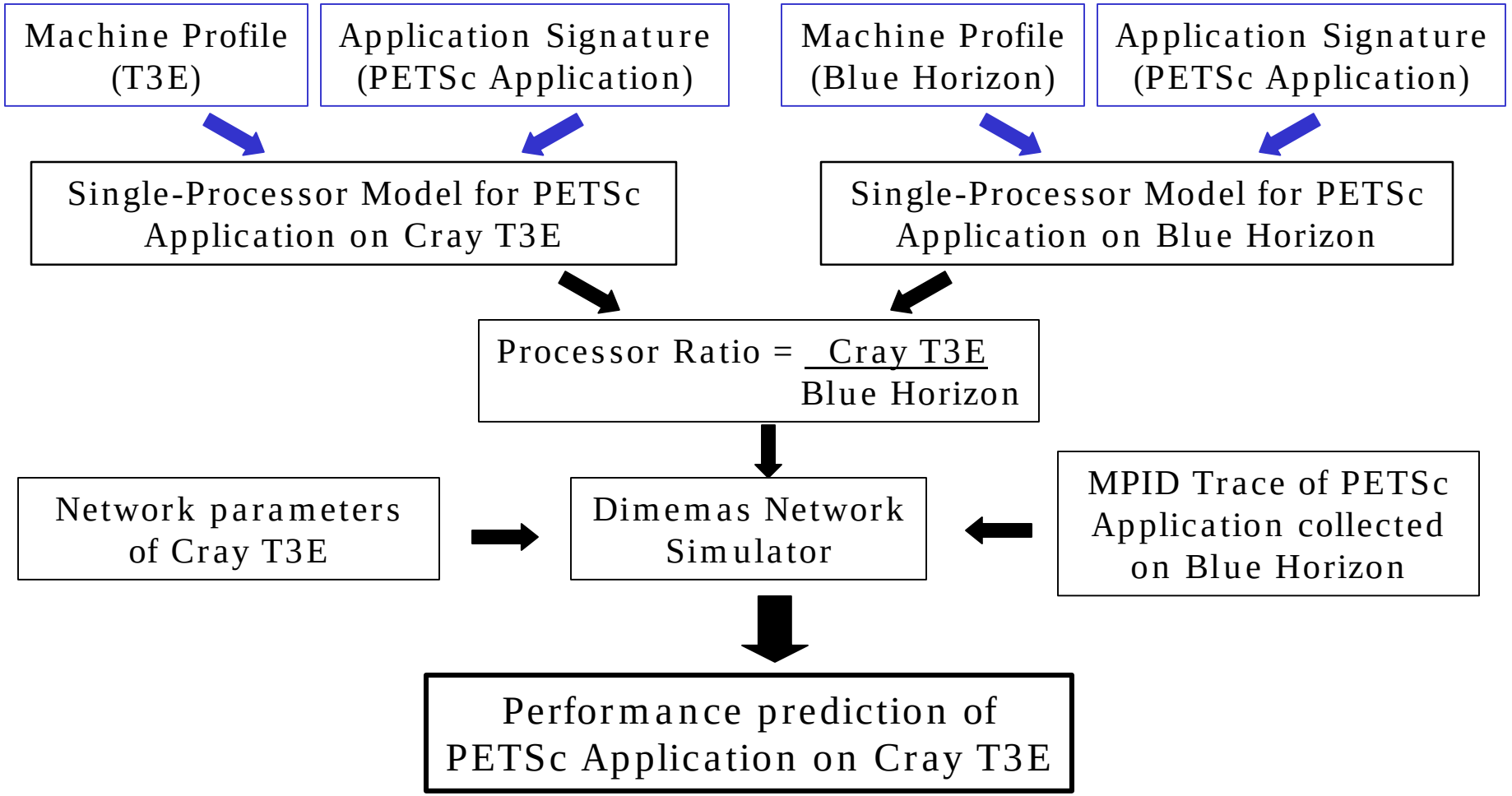
MAPS curve for TCSini for random and non-random loads.

Communications Model – the Network Simulator

- Collect MPI trace of application:
  - Re-link application with MPIDtrace libraries
  - Re-run application – produce MPI trace files
- Network Simulator - DIMEMAS developed at CEPBA see: http://www.cepba.upc.es/
- For prediction of application B on Machine A, need three inputs into simulator :
  1. MPIDtraces of application (collected on machine D)
  2. User defined network parameters (machine A)
  3. Ratio of single-processor performance of machine A to machine D (to model processor performance between communication events i.e Single-Processor Model)

Communications Model – the Network Simulator

Prediction of PETSc Application on the Cray T3E



Performance Prediction of PETSc Kernel

- PETSc – Portable, Extensible Toolkit for Scientific Computation
- Matrix.F – Dense Matrix-vector multiply
- EX19.c – 2-D driven cavity code that uses a velocity-vorticity formulation and finite difference discretization on a structured grid
- MPIDtraces collected on SDSC's Blue Horizon
- Application Signatures (MetaSim Tracer) collected on PSC's TCSini and Lemieux

Performance Predictions – Matrix.F

Predictions for PSC's Lemieux

# CPUs	Real Time	Predicted Time	% Error
2	19.60	22.63	14.33
4	20.36	21.07	3.47
8	20.93	23.66	13.01
64	30.54	31.58	3.38
96	31.84	32.93	3.42
128	34.58	36.81	6.44

Predictions for PSC's TCSini

# CPUs	Real Time	Predicted Time	% Error
2	26.71	27.40	2.58
4	27.63	26.54	3.94
8	27.97	28.65	2.43
64	40.15	38.56	3.97
96	43.77	38.82	11.31
128	49.78	44.37	10.86

Predictions for SDSC's Blue Horizon

# CPUs	Real Time	Predicted Time	% Error
2	31.78	31.82	0.13
4	29.07	31.27	7.57
8	36.13	33.72	6.67
64	44.91	43.91	2.23
96	48.87	47.15	3.52
128	52.88	52.46	0.79

Predictions for TACC's LongHorn

# CPUs	Real Time	Predicted Time	% Error
2	14.95	14.26	4.64
4	14.45	13.92	3.64
8	17.01	15.19	10.68

- Problem size ~100MB/cpu
- Strong scaling of problem size used

Performance Predictions – EX19.c

Predictions for PSC's TCSini

# CPUs	Real Time	Predicted Time	% Error
2	45.00	50.10	11.34
4	35.93	35.12	2.28
8	32.58	28.82	11.55

Predictions for PSC's Lemieux

# CPUs	Real Time	Predicted Time	% Error
2	30.75	32.05	4.23
4	25.18	22.51	10.61
8	20.83	18.51	11.16

Predictions for SDSC's Blue Horizon

# CPUs	Real Time	Predicted Time	% Error
2	66.51	66.59	0.03
4	46.44	46.69	0.55
8	32.40	33.16	2.34

Predictions for TACC's LongHorn

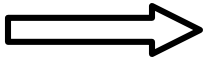
# CPUs	Real Time	Predicted Time	% Error
2	23.83	24.56	3.07
4	18.90	16.78	11.22
8	16.19	13.24	18.22

Prediction of Hardware Upgrades and Future Architectures

- Hardware Upgrades:

  Lemieux & TCSini predictions show that given a processor upgrade and switch upgrade to TCSini, you can predict (avg. error ~7.8%) the benefits to your application.

TCSini:
667 MHz processor
1 rail Quadrics network

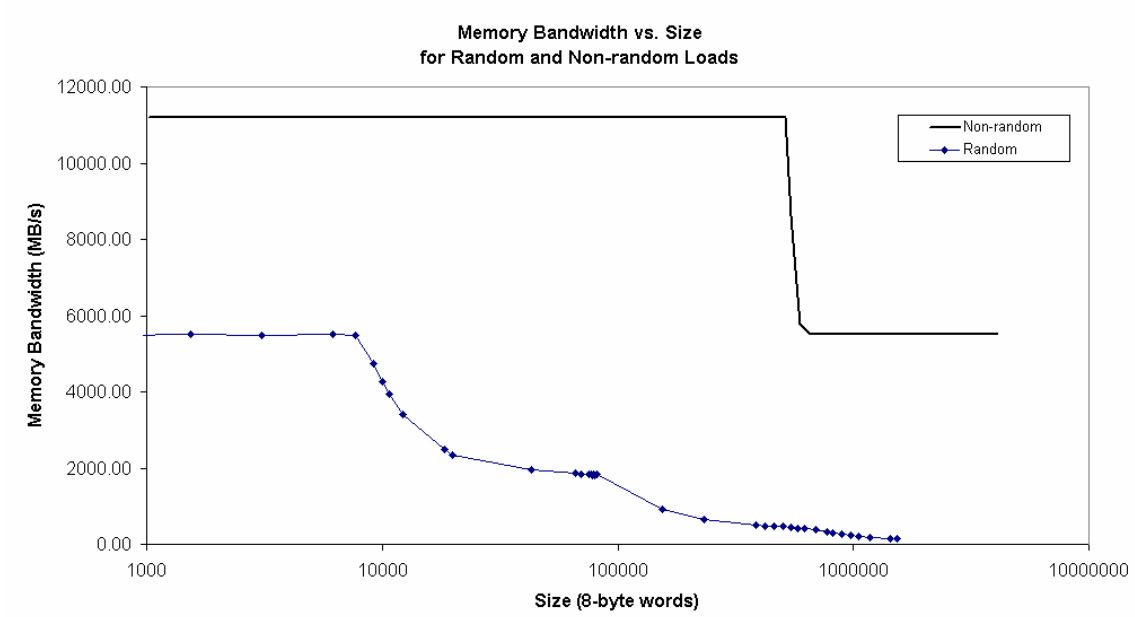


Lemieux:
1000 MHz processor
2 rail Quadrics network

Prediction of Hardware Upgrades and Future Architectures

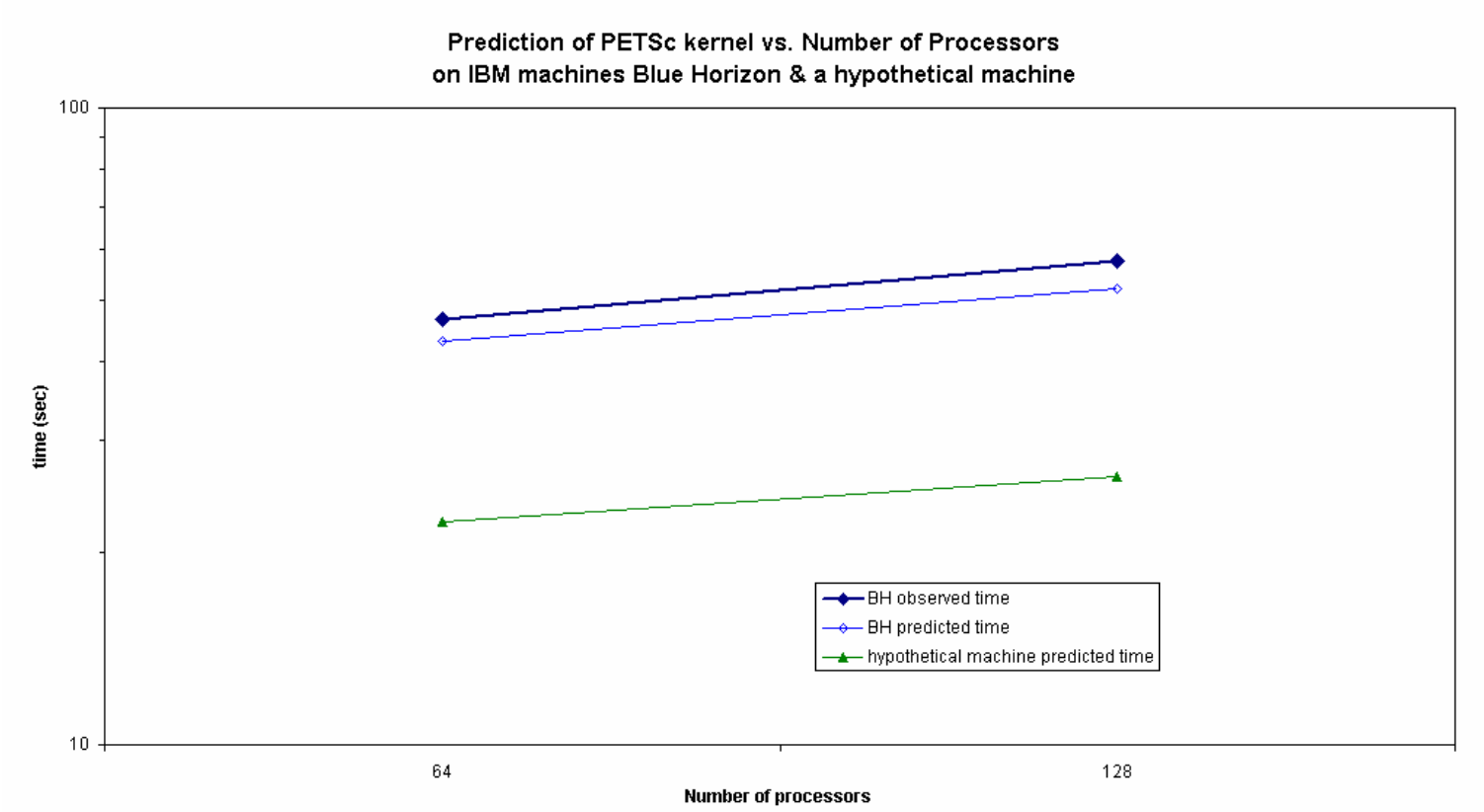
- Future Architectures:

  given an estimation of MAPS curve and network performance, you can predict



Prediction of Hardware Upgrades and Future Architectures

- Future Architectures Prediction:



Extending Predictions to Large Numbers of Processors

- Current research focusing on creation of MPIDtraces for Dimemas Simulator:
  - Extrapolation of MPI patterns
  - Extrapolation of CPU burst between communication events
  - Started with Matrix.F code and working on Linpack benchmark

Extending Predictions to Large Numbers of Processors

- Identifying MPI communication patterns: Example Matrix.F

  Global operations scale:

  Sends as # CPUs increases: 4*N, 2*N,..

  Recv. as # CPUs increases: 8*N

  Where: N is sqrt(size of global array)

  MPIDtrace for 16 CPUs extrapolated from 2 and 4 CPUs ~5.8% error

Conclusions

- Current framework has relatively good accuracy for performance predictions.
- Performance predictions can be useful to:
  - Identify the performance bottlenecks of an application
  - Aid users in allocations request and application porting issues
  - Assist users and centers in determining the benefits of upgrades and future machines on current applications/workloads
  - Aid users understand scalability issues on large numbers of processors

Acknowledgements

This work is sponsored by the Department of Energy Office of Science through SciDAC award "High-End Computer System Performance: Science and Engineering" as part of PERC (Performance Evaluation Research Center perc.nersc.gov)

Computer time was provided by the San Diego Supercomputer Center, Pittsburgh Supercomputer Center, and the Texas Advanced Computing Center